The Traveling Salesman Problem: Low-Dimensionality Implies a Polynomial Time Approximation Scheme

[Preliminary draft]

Yair Bartal*
The Hebrew University

Lee-Ad Gottlieb†
The Hebrew University

Robert Krauthgamer‡
The Weizmann Institute

December 6, 2011

Abstract

The Traveling Salesman Problem (TSP) is among the most famous NP-hard optimization problems. We design for this problem a randomized polynomial-time algorithm that computes a $(1+\varepsilon)$-approximation to the optimal tour, for any fixed $\varepsilon > 0$, in TSP instances that form an *arbitrary* metric space with bounded intrinsic dimension.

The celebrated results of Arora [Aro98] and Mitchell [Mit99] prove that the above result holds in the special case of TSP in a fixed-dimensional Euclidean space. Thus, our algorithm demonstrates that the algorithmic tractability of metric TSP depends on the dimensionality of the space and not on its specific geometry. This result resolves a problem that has been open since the quasi-polynomial time algorithm of Talwar [Tal04].

*Work supported in part by an Israel Science Foundation grant #1609/11. Email: yair@cs.huji.ac.il

†Work supported in part by Israel Science Foundation grants #1109/07 and #1609/11, and by a fellowship from the Lady Davis Fellowship Trust. Email: leead@cs.huji.ac.il

‡Work supported in part by an Israel Science Foundation grant #452/08, a US-Israel BSF grant #2010418, and by a Minerva grant. Email: robert.krauthgamer@weizmann.ac.il

1 Introduction

Among all NP-complete problems, the *Traveling Salesman Problem (TSP)* stands out as fundamental and extensively studied. Indeed, numerous articles and even whole books ([Rei94, LLKS85, GP02, ABCC07]) are devoted to TSP, studying various algorithms for different families of instances. In fact, some of the most basic techniques in combinatorial optimization were devised to tackle TSP, including for instance cutting planes. The input for (the optimization version of) TSP is a complete graph, whose vertex set we denote by $S = [n]$, together with edge-weights $w(\cdot, \cdot)$ that are nonnegative and symmetric,[1] and the goal is to find a closed tour of S of minimum (total) weight, where a tour T is simply a permutation of S, i.e. it visits every vertex exactly once.

A prominent special case of TSP, called *metric TSP*, is where the edge-weights satisfy the triangle inequality,[2] and hence the input is simply a (finite) metric space on the point set $S = [n]$. The importance of this variant lies in the fact that edge-weights arising in many of the typical applications naturally represent lengths and distances. Metric TSP offers some basic structure that may be leveraged by algorithms. In particular, Christofides [Chr76] designed a 1.5–approximation, a polynomial-time algorithm that computes a tour whose weight exceeds the optimum by a factor of at most 1.5. It is a long-standing open problem to improve this approximation for metric TSP, but it is known that there exists a constant $c > 1$, for which c–approximation is NP-hard [PY93, PV06].

Celebrated results of Arora [Aro98] and Mitchell [Mit99] prove that the important special case of metric TSP where the input metric forms a Euclidean metric, admits a PTAS.[3] To be more precise, these PTAS results apply to input metrics that are finite subsets of a *fixed-dimensional* Euclidean metric. Observing that these PTAS results require two separate conditions – Euclidean space and fixed dimensionality – it is only natural to ask:

Question 1.1. *Do TSP instances which satisfy only one of the* bounded dimension *and* Euclidean metric *properties admit a PTAS?*

The bounded-dimensionality requirement turns out to be necessary, as Trevisan [Tre00] shows that TSP in Euclidean metrics (of dimension $\log n$) is NP-hard to approximate within some constant $c > 1$. It is therefore not surprising that the running time of the aforementioned PTAS is doubly-exponential in the dimension.

Eliminating the Euclidean requirement was first addressed by Talwar [Tal04]. Observe that a basic premise of this question is that the notion of dimension applies to an arbitrary (non-Euclidean) metric space. This is indeed possible, and Talwar relied on a definition put forth by Gupta, Krauthgamer and Lee [GKL03] (following [Ass83, Cla99]): The *doubling dimension* of a (finite) metric space S, denoted $\mathrm{ddim}(S)$, is the smallest $k > 0$ such that every ball in the metric can be covered by 2^k balls of half the radius. This definition is essentially based on volume growth, and indeed simple volume estimates imply that a k-dimensional Euclidean metric has doubling dimension $\Theta(k)$. The opposite direction, however, is not true and in fact the family of metrics with bounded doubling dimension is significantly larger than that of bounded-dimensional Euclidean metrics (see [Laa00, LP01, Laa02, GKL03] for details). Talwar [Tal04] generalized much of Arora's machinery [Aro98] and showed that TSP in metrics with fixed doubling dimension admits

[1] Formally, $w(x, y) = w(y, x) \geq 0$ for all $x, y \in S$.

[2] The triangle inequality says that $w(x, y) \leq w(x, z) + w(z, y)$ for all $x, y, z \in S$.

[3] PTAS, which stands for a Polynomial-Time Approximation Scheme, means that for every *fixed* $\varepsilon > 0$ there is a $(1 + \varepsilon)$–approximation. Note that the runtime is polynomial in n for every constant $\varepsilon > 0$.

a QPTAS.[4] But despite repeated attempts, the original goal remained open:

Question 1.2. *Does TSP in metrics of bounded doubling dimension admit a PTAS?*

This question has fascinated researchers for several reasons (see e.g. [Lee10]). First, the existence of a QPTAS may be interpreted as evidence that a PTAS is possible. Second, the above question accords well with a research program that was initiated in [GKL03, KL04, Tal04], and studies the analogy between Euclidean metrics of fixed dimension and general metrics of fixed doubling dimension, from the perspective of algorithmic tractability. Empirically, it has been observed that many algorithms dealing with the former family of metrics can be adapted to deal with the latter, see e.g. [ACGP10, GKK10] and references therein for recent instantiations.

A natural approach to resolving this problem in the positive would be to embed the original metric space in bounded dimensional Euclidean space (such embeddings were studied in [ABN08, ABN11]), and then apply the PTAS of [Aro98, Mit99]. While this general approach has been quite successful in resolving many other algorithmic problems (see for example [Bar96]), it fails here since any such embedding must have non-constant distortion ([Laa00, LP01, Laa02]), in fact $\Omega(\sqrt{\log n})$ ([GKL03]). It appears that achieving a PTAS for arbitrary bounded-dimensional metric spaces requires a new approach to bypass the limitations of the embedding.

1.1 Results

Our central contribution is a PTAS for TSP in metrics of fixed doubling dimension.

Theorem 1.3. *A $(1+\varepsilon)$-approximation to the optimal tour of a metric TSP instance S on $n = |S|$ points can be computed by a randomized algorithm in time* $n^{2^{O(\mathrm{ddim}(S))}} \cdot 2^{(2^{\mathrm{ddim}(S)}/\varepsilon)^{O(\mathrm{ddim}(S))}\sqrt{\log n}}$.

The previously known running time is is quasipolynomial in n, namely $2^{(\mathrm{ddim}(S)/\varepsilon \cdot \log n)^{O(\mathrm{ddim}(S))}}$, due to Talwar [Tal04, Theorem 8].

1.2 Techniques

We build upon the framework of [Aro98, Tal04], and introduce two main new ideas (and several more minor ones). Our framework is a variant of Talwar's algorithm (see Section 2) that includes: (1) a randomized hierarchical clustering (partitioning) of S; (2) the introduction of portals around every cluster; (3) slightly modifying the optimal tour (for sake of analysis only) so that the tour is portal-respecting (crosses every cluster only at its portals) and has few crossings into the cluster; (4) a dynamic program that computes a tour for each cluster based on the tours already computed for its subclusters.

Our first new idea (in Section 1.5) is to estimate the cost incurred by an optimal tour inside a ball. Intuitively, the estimate is merely an instantiation of the well-known 2–approximation of TSP using the minimum spanning tree (stated as Lemma 1.6). But in reality, edges entering and exiting the ball interfere with this calculation, and thus the estimate includes both multiplicative and additive error terms.

Our second new idea is to treat separately dense regions in the metric space, meaning balls in which an optimal tour incurs a relatively large cost. Now if all regions are sparse, then we are

[4] QPTAS, which stands for a Quasi-Polynomial Time Approximate Scheme, means that for every *fixed* $\varepsilon > 0$ there is a $(1+\varepsilon)$–approximation running in quasi-polynomial time $2^{\mathrm{polylog}(n)}$.

almost done – in this case we use limited randomization to determine the hierarchical clustering. In particular, we draw at random $O(\log n)$ radii-values for every cluster center, and argue that with high probability at least one of them is useful for the construction of a good partition. We then augment the aforementioned dynamic program to search also over the different radii-values for every cluster center. (This appears in Section 3.1.) If there is a dense region, then we can use the first idea above to find the (nearly) smallest one. We then "split" the TSP instance into two portions, roughly the inside and the outside of that dense region. The outside is solved recursively. The inside portion is nearly sparse because it can be covered by a limited number of smaller (and thus sparse) balls, and so it can be solved immediately by the sparse algorithm. Stitching the solutions for the two portions may be costly, but since the region is dense, we can effectively charge our algorithm's cost to that of the optimum. (This appears in Section 3.2.)

1.3 Related work

A few hardness of approximation results are known. That general (not necessarily metric) TSP is NP-hard follows immediately from Karp's original NP-hardness proof for Hamiltonian cycle [Kar72]. Moreover, this proof shows that TSP does not admit any finite factor approximation in polynomial time, unless P=NP. Papadimitriou and Yannakakis [PY93] showed that metric TSP is hard to approximate within some constant factor $c > 1$, even if all the metric distances are either 1 or 2. Papadimitriou and Vempala [PV06] proved that approximating metric TSP within factor 220/219 is NP-hard. Papadimitriou [Pap77] showed that two-dimensional Euclidean TSP is NP-hard.

The runtime of Arora's algorithm [Aro98] was later improved in [RS98], and his geometric approach was subsequently employed for other Euclidean problems in [CL98, ARR98, CLZ02, KR07]. Further extention of the algorithms of [Aro98, Tal04] to the problem of TSP with neighborhoods (under mild conditions) include [Mit07] and [CE11]. Chan and Gupta [CG08] gave an algorithm for TSP that runs in sub-exponential time in metrics in which an alternative (more general) notion of dimension is assumed to be bounded.

1.4 Preliminaries

Recall our notation for the metric TSP instance: S denotes the set of points, $d(\cdot,\cdot)$ their pairwise distances, $\mathrm{ddim}(S)$ its doubling dimension, and $n = |S|$ its size. We fix $0 < \varepsilon < 1/20$, which determines the desired approximation to be $1+O(\varepsilon)$. We may assume that $\varepsilon > 1/n$, as otherwise all our results hold trivially — TSP can be solved exactly in time $n!$ by straightforward enumeration, providing better approximation and faster runtime than our claimed runtime (which is at least exponential in $1/\varepsilon$). Similar to what was done by Arora [Aro98], by suitable scaling we may assume that the minimum interpoint distance in S is 1 and the diameter is $O(n/\varepsilon) = O(n^2)$.

As usual, the metric may be viewed as the complete graph on S, with edge weights corresponding to pairwise distances, denoted $w(x,y) \stackrel{\text{def}}{=} d(x,y)$. A set of points $S' \subseteq S$ is sometimes called a *cluster*. We let $\mathrm{MST}(S')$ denote a minimum spanning tree (breaking ties arbitrarily) of the complete graph on S'. The *ball* centered at $x \in S$ with radius $R > 0$ is defined as $B(x,R) \stackrel{\text{def}}{=} \{y \in S :\ d(x,y) \le R\}$. We define $B^*(x,R)$ to be the edges of the complete graph on $B(x,R)$.

Tours. Throughout, a *tour* is a finite sequence of points; by convention, it is undirected. A transition in the tour is a pair of successive points in the sequence, which may be viewed as an edge

in the complete graph on S. A *closed* tour is defined in the natural way by adding a transition between its two endpoints (and then no point is an endpoint in the tour).

The *weight* of a multiset M of transitions is defined as $w(M) \stackrel{\text{def}}{=} \sum_{(x,y)\in M} w(x,y)$. This notation naturally extends to a tour T, by viewing T as sequence of transitions, and so $w(T)$ just denotes the (total) length of the tour T.

Let $\mathrm{OPT}(S')$ denote a minimum length closed tour that visits all points of $S' \subseteq S$.

Lemma 1.4. *If a tour T traverses an edge e more than twice, then there exists a lighter tour T' that is a subset of the transitions of T.*

Proof. We will prove the case of three traversals, and a similar proof holds for more. Suppose that T traverses some edge $e = (u, v)$ exactly thrice. Then an ordering of the edges of T must take the form $E_1eE_2eE_3eE_4$, where each E_i is a (possibly empty) set of edges, and e may be traversed in either direction. If E_2 or E_3 are empty, then the tour is of the form $E_1eeE_3eE_4$ or $E_1eE_2eeE_4$, and the segment ee may be deleted. Now, if E_2 is a cycle beginning and ending at u (or v), then one of $E_1eeE_3E_2eE_4$ or $E_1eeE_3eE_2E_4$ is a valid reordering of T, and the segment ee may be deleted, resulting in a lighter tour. Assume then that E_2 begins at u and ends at v, and trivially E_3 also begins at u and ends at v. Then the segment E_2eE_3e which begins and ends at u may be replaced by a path that follows E_2 and then traverses the edges of E_3 in a backwards order, thereby skipping e twice. □

Doubling dimension. Let $\lambda_S > 0$ be the *doubling constant* of the metric S, the smallest value such that every ball in S can be covered by λ_S balls of half the radius. Recall that the *doubling dimension* of S is $\mathrm{ddim}(S) = \log_2 \lambda_S \geq 1$ (assuming $|S| \geq 2$). The following packing property can be demonstrated via repeated applications of the doubling property (see e.g. [GKL03]).

Lemma 1.5. *(Packing Property) Let $S' \subseteq S$ have minimum interpoint distance $\alpha > 0$. Then*

$$|S'| \leq \left(\frac{2\,\mathrm{diam}(S')}{\alpha}\right)^{\mathrm{ddim}(S)}.$$

Notice that whenever $\frac{\mathrm{diam}(S')}{\alpha} \geq 2$, we can further bound $|S'| \leq \left(\frac{\mathrm{diam}(S')}{\alpha}\right)^{2\,\mathrm{ddim}(S)}$.

Nets. Similar to what was described in [GGN06, KL04], a subset $S' \subseteq S$ is an (a, b)-*net* of S (for $0 < a \leq b$) if it satisfies the following two properties.

(i). Packing: For every $u, v \in S$ we have $d(u, v) > a$.

(ii). Covering: Every $v \in S$ is within distance b of some point $u \in S'$, i.e., $S \subseteq \cup_{u\in S'} B(u, b)$.

We say that $u \in S'$ *covers* $v \in S$ if $d(u, v) \leq b$. The previous conditions require that the points of S' be spaced out, yet nevertheless cover all points of S. If $a = b$, we may refer to the (a, b)-net as an a-net.

Hierarchy of nets (or point hierarchies). Set $L \stackrel{\text{def}}{=} \lceil \log_s \mathrm{diam}(S) \rceil = O(\log_s n)$, for a parameter $s \geq 4$. (In Section 3 we will require s roughly $(\log n)^{1/\,\mathrm{ddim}(S)}$.) For each $i = 0, \ldots, L$, fix $H_i \subseteq S$ to be an s^i-net of S, called the i-level net, or the s^i-scale net. Using a simple greedy construction, we may assume that $H_i \subseteq H_{i-1}$. Notice that the bottom level $i = 0$ contains all points, and the top level $i = L$ contains only a single point.

Spanning trees, tours, and patching. It is well-known that the optimal tour on a set S' is approximated within factor 2 by the minimum spanning tree on S'.

Lemma 1.6. *Let $S' \subseteq S$. Then $w(\mathrm{MST}(S')) \leq w(\mathrm{OPT}(S')) \leq 2w(\mathrm{MST}(S'))$.*

The following lemma, due to Talwar [Tal04] (see also [Smi10]), uses the doubling dimension to bound $w(\mathrm{MST}(S))$.

Lemma 1.7. *Let $S' \subseteq S$. Then $w(\mathrm{MST}(S')) \leq 4|S'|^{1-1/\mathrm{ddim}(S)} \cdot \mathrm{diam}(S')$.*

The following lemma, due to [Aro98, Tal04], is known as the Patching Lemma for doubling spaces. We say that a transition (x, y) in a tour T *crosses* a cluster $C \subseteq S$ if exactly one of x, y belongs to C. The point (among x, y) that belongs to C is called a cross-point.

Lemma 1.8 (Patching Lemma)**.** *Let T be a tour that crosses a cluster C r times. Then the number of crossings can be reduced to two, at an additional cost of at most 4 times the minimum spanning tree of T's cross-points (denoted $\hat{C}$). That is, there is a tour T' that crosses C at most twice and (plugging in Lemma 1.7)*

$$w(T') \leq w(T) + 4w(\mathrm{MST}(\hat{C})) \leq w(T) + 16r^{1-\frac{1}{\mathrm{ddim}(S)}} \mathrm{diam}(C).$$

We also present the following lemma, which is a version of the Patching Lemma tailored to our specfic needs.

Lemma 1.9. *Let T be a tour of S, and $C \subset S$ be a cluster. Consider the tour segments T_i $i = 1, \ldots, k$ that result from removing from T edges not found in the full graph on C, and let $\hat{C}$ be the cross-points of C. Then there exists a closed tour T' on C which contains only edges in T_i and $\mathrm{MST}(\hat{C})$ and*

$$w(T') \leq 4w(\mathrm{MST}(\hat{C})) + \sum_{i=1}^{k} w(T_i) \leq 16r^{1-\frac{1}{\mathrm{ddim}(S)}} \mathrm{diam}(C) + \sum_{i=1}^{k} w(T_i).$$

Let $C \subset S$, and let $\hat{C}$ be the cross-points of C. We say that a tour T *exits* C if it transitions from a point of C to a point of $\hat{C}$ (the exit point) and then to a point outside the cluster. Similarly, we say that a tour T *enters* C if the tour transitions from a cross-point in $\hat{C}$ (the entry point) to another point of C, after having immediately arrived at the cross-point from a point outside C.

Lemma 1.10. *Given any tour T, there exists a tour T' which is a subset of the transitions of T, such that T' enters and exits all disjoint clusters at most twice at each cross-point.*

Proof. The proof is an easy application of Lemma 1.4. For each cluster C, represent each crosspoint x as two points x_1 and x_2 with infintesimally small distance between them. Let edges connecting x to points outside C be incident on x_1, and edges connecting x to points of C be incident on x_2. Then edge $e = (x_1, x_2)$ is traversed at most twice. □

Net-Respecting Tours. A tour is said to be *net respecting (NR)* if every transition in it, say of length ℓ, has both of its endpoints belonging to every net H_i with $s^i \leq \varepsilon\ell$. (This definition is with respect to a given hierarchy $\{H_i\}$ and $\varepsilon > 0$.) Since $\mathcal{H}$ is a hierarchy, it actually suffices for the endpoints to belong to H_i for the maximum i such that $s^i \leq \varepsilon\ell$.) We denote by $\mathrm{OPT}^{NR}(S')$ an optimal (minimum length) net-respecting tour that visits all points of $S' \subseteq S$.

Lemma 1.11. *Every tour T can be converted to a net-respecting tour T' such that $w(T') \leq (1 + 6\varepsilon)w(T)$ and T' visits all points visited by T.*

Proof. Simply replace a transition (x, y) with (x', y'), where x', y' are the i-net points covering x, y respectively (where level i is the highest level such that $s^i \leq \frac{\varepsilon}{2}d(x, y)$), and connect x to x' (and similarly y to y') via a sequence of net points (from levels $j = i - 1, i - 2, \ldots$) covering x. Clearly $d(x', y') \leq (1 + \varepsilon)d(x, y)$, and the path from x to x' (or y to y') has length at most $\sum_{j \leq i} s^j \leq 2s^i \leq 2\varepsilon d(x, y)$. □

1.5 Local behavior of optimal tour

We next show that the weight of the optimal (net-respecting) tour inside some neighborhood can be approximated using a minimum spanning tree of points in that neighborhood.

Lemma 1.12. *Let $\mathrm{OPT}^{NR}(S)$ be an optimal net-respecting tour visiting all points in S. Then*

(i). $w(\mathrm{OPT}^{NR}(S) \cap B^*(u, R)) \leq 6(1 + 6\varepsilon) \cdot w(\mathrm{MST}(B(u, R)))$.

(ii). $w(\mathrm{MST}(B(u, R))) \leq w(\mathrm{OPT}^{NR}(S) \cap B^*(u, 4R)) + (s/\varepsilon)^{2\,\mathrm{ddim}(S)}R$.

Proof of Lemma 1.12. Assume by contradiction that (i) does not hold. Applying the Patching Lemma to the tour $\mathrm{OPT}^{NR}(S)$ with respect to the cluster $B(u, R)$, we get a modified tour which visits all of S and crosses that cluster at most twice, while increasing the tour's length by at most $4\,\mathrm{MST}(B(u, R))$. Now replace the portion of this tour between these unique cross-points with a tour that is derived from an MST of $B(u, R)$ (Lemma 1.7), and thus adds total length of at most $2w(\mathrm{MST}(B(u, R)))$. Convert the resulting tour to be net-respecting (Lemma 1.11); this step clearly affects only the newly-added edges. This entire process first removes from the tour a total length of at least $w(\mathrm{OPT}^{NR}(S) \cap B^*(u, R))$, then adds a total length of at most $6(1+6\varepsilon) \cdot w(\mathrm{MST}(B(u, R)))$. But since (i) does not hold, it means the overall tour length has strictly decreased, which contradicts the optimality of $\mathrm{OPT}^{NR}(S)$.

To prove part (ii), consider a tour $\mathrm{OPT}^{NR}(S)$. It can be partitioned into subtours $T_1, T_2, T_3, \ldots$, where T_k for odd k contains only points in $B(u, R)$ and is maximal with respect to containment. Thus, T_k for even k has its endpoints inside $B(u, R)$, overlapping T_{k-1} and T_{k+1}, and the remaining points in T_{k-1} and T_{k+1} are outside $B(u, R)$. By definition, $w(\mathrm{OPT}^{NR}(S)) = \sum_k w(T_k)$.

We will now construct a connected graph that spans $B(u, R)$, and use a charging argument to bound its weight. First, take $T_1, T_3, T_5, \ldots$; we can pay for them using their own contribution to $\mathrm{OPT}^{NR}(S) \cap B^*(u, 4R)$. Now consider any T_{2k}. If this T_{2k} visits only points inside $B(u, 4R)$, then again take T_{2k} itself, paying for it using its own contribution to $\mathrm{OPT}^{NR}(S) \cap B^*(u, 4R)$. Otherwise, we know that T_{2k} crosses the ball $B(u, 4R)$, and obviously this happens (at least) twice. If one of the two corresponding cross-points is inside $B(u, 4R) \setminus B(u, 3R)$, then this T_{2k} contributes to $\mathrm{OPT}^{NR}(S) \cap B^*(u, 4R)$ at least $2R$, which suffices to pay for connecting T_{2k-1} and T_{2k+1} with a direct edge.

The final case is when T_{2k} crosses the ball $B(u, 4R)$, obviously (at least) twice, and in both crossings the cross-point is inside $B(u, 3R)$, implying that this transition has length at least R. Let i be the largest value for which $s^i \leq \varepsilon R$. By the net-respecting property, the cross-point must belong to an s^i-net. Although we may have many such subtours T_{2k}, the number of distinct s^i-net points inside $B(u, 3R)$ is at most $(\frac{3R}{\varepsilon R/s})^{\mathrm{ddim}(S)} \leq \frac{1}{2}(s/\varepsilon)^{2\,\mathrm{ddim}(S)}$, and we then connect all these net points to an arbitrary point, say in T_1, at a cost of at most $(s/\varepsilon)^{2\,\mathrm{ddim}(S)} \cdot R$. We also take the portion of T_{2k} that goes until these cross-points (which are also the said net-points), paying for this portion using T_{2k}'s own contribution to $\mathrm{OPT}^{NR}(S) \cap B^*(u, 4R)$. Overall, the edges that we take are easily seen to form a connected subgraph that spans all vertices of $B(u, R)$ and has total weight at most $w(\mathrm{OPT}^{NR}(S) \cap B^*(u, 4R)) + (s/\varepsilon)^{2\,\mathrm{ddim}(S)}R$; thus, the weight of the MST on these points cannot be larger. □

2 TSP via hierarchical clustering (Arora and Talwar)

As an exposition to our PTAS, we review a variant of the algorithm of Talwar [Tal04] (and in turn Arora [Aro98]), which uses hierarchical clustering to compute a $(1+\varepsilon)$-approximate tour in quasi-polynomial time. Recall that we may assume (by arguments found in [Aro98, Section 2.1.1]) that the instance of TSP is a set S with minimum interpoint distance 1 and diameter $O(n/\varepsilon) = O(n^2)$. The construction uses a hierarchy of nets as described above. We first introduce the single-scale partition invoked by the algorithm. This partition follows the same framework used in [Bar96, Bar98, FRT03, GKL03, ABN11], and is slightly different from the one that appeared in [Tal04] in that it uses the exponential distribution.

Single-scale probabilistic partition. Fix a set $S' \subseteq S$ to be partitioned. Fix a level i, and impose an arbitrary ordering π on the points of the s^i-net $H_i \subseteq S$. The clusters are formed one by one following the ordering π. Each point of H_i constitutes a cluster center. With each net-point $u \in H_i$ we associate a random radius $h_u \in [s^i, 2s^i]$ from an exponential distribution.[5] The ball $B(u, h_u)$ constitutes a new cluster of S', which is removed and then the process continues to form the rest of the clusters. The boundary of u's cluster (i.e. edges that cross the cluster) is determined only by the ordering imposed by π, and by the balls associated with cluster centers at distance at most $4s^i$ from u. By the packing property, there are at most $2^{4\,\mathrm{ddim}(S)}$ such cluster centers.

The next claim follows from [ABN11].

Claim 2.1. *For every $u, v \in S' \subseteq S$, the probability that the single-scale probabilistic partition assigns u and v to different clusters (they are* cut*) is at most $\frac{c'\,\mathrm{ddim}(S) d(u,v)}{s^i}$ for some absolute constant $c' > 0$.*

Hierarchical clustering. To create the clustering, we first choose a single-scale partition for the top level L. As described above, each net point chooses a radius in the range $[s^L, 2s^L]$, and then every point in S is assigned to the earliest ball that covers it. For the next hierarchical level $L-1$, we take each L-level cluster separately, and build for its points a new partition with random radius in the range $[s^{L-1}, 2s^{L-1}]$. (We may assume for simplicity that the partition employed inside each

[5] The density function of the distribution can take the form: $\frac{2^{8\,\mathrm{ddim}(S)}}{1-2^{-8\,\mathrm{ddim}(S)}} \cdot \frac{8 \ln 2\,\mathrm{ddim}(S)}{s^i} \cdot 2^{-\frac{8\,\mathrm{ddim}(S)}{s^i} r}$, where $r \in [s^i, 2s^i]$, and 0 for all other values of r (see [ABN11]).

L-level cluster is the same, i.e. it uses the same ordering and random radii, although the analysis does not require this assumption.) The construction continues recursively until level 0, the bottom level. Note that by the packing property (Lemma 1.5), each cluster has at most $s^{2\,\mathrm{ddim}(S)}$ child clusters.

The probability that u and v are cut at level i (found in different i-level clusters) is bounded by the sum of the probabilities that they are cut in any level i or higher, that is $\sum_{j=i}^{L} \frac{c'\,\mathrm{ddim}(S) d(u,v)}{s^j} = O\left(\frac{\mathrm{ddim}(S) d(u,v)}{s^i}\right)$.

TSP algorithm and analysis. The dynamic programming TSP algorithm functions on the hierarchical clustering above. A tour is (m,r)-light with respect to the hierarchical partition if it crosses each i-level cluster at most r times, and only at $\frac{s^i}{M}$-net points, the cluster *portals*, where m is an upper bound on the number of portals. We choose $s \geq 4$, and let M be the smallest power of s that is greater or equal to $\frac{\mathrm{ddim}(S)\log n}{\varepsilon}$, so $m \leq M^{2\,\mathrm{ddim}(S)} \leq \left(\frac{s\,\mathrm{ddim}(S)\log n}{\varepsilon}\right)^{2\,\mathrm{ddim}(S)}$.

An optimal (m,r)-light tour for the hierarchical clustering can be computed by dynamic programming as follows: Consider a cluster C. Any valid (m,r)-light tour crosses C at most r times and only at portals, so it consists of r paths starting and ending at portals. A *configuration* is a multiset of r or fewer portals partitioned into pairs (each representing an entry/exit pair). A single portal may appear more than once in the configuration if the tour crosses it multiple times, but each instance counts towards r. A cluster has m portals, so there are no more than m^r possible configurations. Now, if optimal (m,r)-light tours have been inductively computed for the (at most) $s^{2\,\mathrm{ddim}(S)}$ children of C under the hierarchical clustering, the optimal (m,r)-light tour for C can be computed by a brute-force algorithm: Since the (m,r)-light tour of each child cluster enters and exits via a portal, we can "stitch" together the child tours through the child portals. For each fixed configuration of C (at most m^r possible configurations), we consider all possible child configurations (at most $m^{s^{2\,\mathrm{ddim}(S)}r}$). Having fixed a configuration for every child cluster, we have at most $s^{2\,\mathrm{ddim}(S)}r$ candidate child portals where the tour may cross. Since each child portal may be connected to one of at most $s^{2\,\mathrm{ddim}(S)}r$ other candidate child portals, all possible graphs connecting these portals can be enumerated in time bounded by $\left(s^{2\,\mathrm{ddim}(S)}r\right)^{s^{2\,\mathrm{ddim}(S)}r} \leq r^{s^{4\,\mathrm{ddim}(S)}r}$. Then we consider all graphs that support a tour connecting all child portals to portals of C; we choose the graph with the least cost tour. The total runtime is $(mr)^{s^{O(\mathrm{ddim}(S))}r}$.

Crucially, it follows from [Tal04] that with constant probability, the hierarchical clustering for S admits an (m,r)-light tour with weight at most $(1+\varepsilon)\,\mathrm{OPT}(S)$, for $s = 4$, and $m, r \stackrel{\mathrm{def}}{=} \left(\frac{\mathrm{ddim}(S)\log_s n}{\varepsilon}\right)^{2\,\mathrm{ddim}(S)}$. The proof proceeds as in [Aro98], by showing that an optimal tour can be slightly modified to observe this property. The cost of modifying the tour is charged to the tour's edges, and the analysis shows that the cost charged to each edge is small. Briefly, the probability that an edge $e = (u,v)$ is cut by the i-level partition is bounded by $\frac{c' d(u,v)\,\mathrm{ddim}(S)}{s^i}$. We then move this edge to be incident on a $\frac{s^i}{M}$-net point, at an additive cost (increase in tour length) of $\frac{2s^i}{M}$. Hence, the expected cost of moving e due to a cut at level i is $\frac{c' d(u,v)\,\mathrm{ddim}(S)}{s^i} \cdot \frac{2s^i}{M} = O\left(\frac{\varepsilon d(u,v)}{\log_s n}\right)$, and the expected cost of moving e due to a cut in any of $O(\log_s n)$ levels is $O(\varepsilon \cdot d(u,v))$. Now, if the optimal tour crosses an i-level cluster some $r_o \geq r$ times, the tour is patched via the minimum spanning tree on the cross points (via Lemma 1.8). The cost is charged to the edges participating in the patching, at a per edge cost of $O\left(\frac{s^i r_o^{1-1/\mathrm{ddim}(S)}}{r_o}\right) = O\left(\frac{s^i \varepsilon}{\mathrm{ddim}(S)\log_s n}\right)$. But an edge participates

in a patching only if it is cut, which happens with the above probability, hence the expected charged cost to e due to patchings at one level is $O\left(\frac{d(u,v)\,\mathrm{ddim}(S)}{s^i} \cdot \frac{s^i\varepsilon}{\mathrm{ddim}(S)\log_s n}\right) = O\left(\frac{\varepsilon \cdot d(u,v)}{\log_s n}\right)$, and due to patchings at all levels is $O(\varepsilon \cdot d(u,v))$. The values for m and r imply that the algorithm above runs in quasi-polynomial time.

3 Obtaining a PTAS

In this section, we prove Theorem 1.3, the central contribution of this paper.

Our algorithm for TSP again mimics the one employed by Arora [Aro98]. His algorithm requires a hierarchical partition, yet we cannot directly employ the partition of Section 2. That clustering essentially decides the cluster assignment for each level separately, and hence it cannot successfully invoke the analysis of [Aro98] to bound the expected cost of patchings per level. More precisely, the event that edge e is cut by an i-level single-scale partition, and by no other single-scale partition, is still $\Theta\left(\frac{\mathrm{ddim}(S)}{s^i}\right)$. Hence, the expected cost of participating in an i-level patching is mostly independent of the expected cost of participating in a j-level patching for all $i \neq j$, and so a term of $L = O(\log_s n)$ must appear in r. This is precisely the reason why the analysis presented by Talwar [Tal04] does not achieve a PTAS for metric TSP.

Instead, we will employ a modified version of the above partition, and analyze its performance on net-respecting tours. We will show that if a tour obeys some edge-sparsity property, then it admits an (m, r)-light tour on a hierarchy very similar to the one above. Crucially, the edge-sparsity property allows us to achieve $r = O((\log n)^c)$ for a small constant $c < 1$, which implies a polynomial runtime. (Although we fix the value of c in the analysis, it can in fact be taken as an arbitrarily small constant.) This partition can be found by a "brute-force" version of the above dynamic programming algorithm. We then show that if the tour has an edge-dense area, then the point set S can be broken into two pieces, and TSP solved separately on each.

While a regular optimal tour need cross a point only twice, in an optimal net-respecting tour a net-point may have many edges incident upon it. Because of this, we will slightly modify the term (m, r)-light to mean that tour exits or enters the cluster at most r times at the portals, although there may be many crossings incident on a portal. Recall from Lemma 1.10 that we may assume that a tour enters or exits the internal cluster points at most twice via a single portal.

3.1 An algorithm for sparse tours

A tour T is said to be *q-sparse* with respect to a net-respecting hierarchy $H_1, \ldots, H_L$ if for all $i \in [L]$ and $u \in H_i$, the edges of T inside the ball $B(u, 3s^i)$ have weight $w(T \cap B^*(u, 3s^i)) \leq qs^i$. The last inequality will be referred to as q-sparsity of that ball.

Suppose that an oracle had informed us that S admits a net-respecting tour that is a $(1+\varepsilon)$-approximation to $\mathrm{OPT}(S)$ and is q-sparse. (An oracle with a similar capability is presented in Section 3.2, for $q = (s/\varepsilon)^{O(\mathrm{ddim}(S))} \cdot 2^{O(\mathrm{ddim}^2(S))}$. Nevertheless, the following lemmas are stated for general q.) Then we can prove the following lemma.

Lemma 3.1. *Suppose S admits a net-respecting q-sparse tour T. Then there exists a hierarchical clustering for S which admits an (m, r)-light tour T' with $w(T') \leq (1+\varepsilon)w(T)$ for $m := (s\,\mathrm{ddim}(S)/\varepsilon \cdot \log_s n)^{2\,\mathrm{ddim}(S)}$ and $r = r(q) := 10 \cdot 2^{4\,\mathrm{ddim}(S)} q \log_s \log n + (2c'\,\mathrm{ddim}(S)/\varepsilon)^{\mathrm{ddim}(S)} + (s/\varepsilon)^{2\,\mathrm{ddim}(S)}$.*

We remark that the tour T' need not be net-respecting.

Proof. Fix T. The hierarchical clustering closely follows the description from Section 2, with the only difference being that the cluster radii are chosen a little more carefully. Consider a net-point $u \in H_j$. Because of the q-sparsity of the ball $B(u, 3s^j)$, the weight of edges in T that have length at most s^j and at least one endpoint inside $B(u, 2s^j)$ is at most qs^j. Recall that we wish to assign u a random radius $h_u \in [s^j, 2s^j]$. Let V be the set of values which cut fewer than $10q\,\mathrm{ddim}(S)$ edges of T of length at most s^j. A simple averaging calculation shows that at least a fraction $1 - \frac{1}{10\,\mathrm{ddim}(S)}$ of radii in $[s^j, 2s^j]$ belong to V (fraction here means with respect to uniform distribution). We choose h_u randomly from an exponential distribution on $[s^j, 2s^j]$ [ABN11], and resample until finding a $h_u \in V$. Note that the exponential distribution implies that the probability that a sampled radius belongs to V is at least $1/2$ (this can be easily seen by considering the "worst-case" scenario where $V \subseteq [(1 + \frac{1}{10\,\mathrm{ddim}(S)})s^j, 2s^j]$). Then the clustering is done exactly as before (iterating over centers etc.), but clearly the actual number of edges cut can only be smaller (because of other balls cut earlier in the same level or at a higher level). Note that knowledge of T was necessary only to determine which radii are valid choices for h_u.

We now analyze the expected cost of converting the tour T to be (m, r)-light with respect to this hierarchical clustering. We first consider the cost of forcing the tour to cross every cluster only through its m cluster portals. The probability that an edge $e = (u, v) \in T$ is cut by the i-level partition is bounded by $\frac{2c'd(u,v)\,\mathrm{ddim}(S)}{s^i}$ (the probability that the edge is cut conditioned on choosing a valid radius). We then move this edge to be incident on a $\frac{s^i}{M}$-net point, where M is the smallest power of s at least $\frac{\mathrm{ddim}(S)\log_s n}{\varepsilon}$. The cost of this modification is $\leq \frac{2s^i}{M}$. So the expected cost of moving e due to a cut in level i is at most $\frac{2c'd(u,v)\,\mathrm{ddim}(S)}{s^i} \cdot \frac{2s^i}{M} \leq O\left(\frac{\varepsilon d(u,v)}{\log_s n}\right)$, and the expected cost of moving e due to cuts in all $L = O(\log_s n)$ levels is $O(\varepsilon d(u, v))$. The number of portals is at most $m := \left(\frac{12s^i}{s^i/M}\right)^{\mathrm{ddim}(S)} \leq \left(\frac{12s\,\mathrm{ddim}(S)\log_s n}{\varepsilon}\right)^{\mathrm{ddim}(S)}$ by the packing property (Lemma 1.5).

We turn to the analysis of reducing the number of entrance and exit points to r via patching. Consider some i-level cluster C. We define an edge to be *long* with respect to level i if its length is at least s^i, and otherwise we consider it *short*. We break the analysis into two parts, and bound the number of crossings due to long and short edges separately. It turns out that long edges don't figure into the patchings.

We analyze short edges first. Recall that in the construction of the clustering, each j-level ball cuts at most $10q\,\mathrm{ddim}(S)$ edges of length at most s^j. Further, edges crossing our cluster C could have actually been cut by any of at most $2^{3\,\mathrm{ddim}(S)}$ neighboring balls at any level $j \geq i$ (because there are at most $2^{3\,\mathrm{ddim}(S)}$ j-level balls within distance $4s^j$ from C's center). Therefore the number of short edges of length at most s^i crossed by C due to j-level ball cuts (for any $j \geq i$) is at most $2^{3\,\mathrm{ddim}(S)}\,\mathrm{ddim}(S)q$. Set $r' = r'(q) := \max\{20 \cdot 2^{3\,\mathrm{ddim}(S)}q\,\mathrm{ddim}(S)\log_s \log n, \left(\frac{2c'\,\mathrm{ddim}(S)}{\varepsilon}\right)^{\mathrm{ddim}(S)}\}$ and consider the case where C cuts more than r' short edges. Then at least $r'/2$ of the short edges crossing C must have been cut by balls in levels $j' \geq i + 2\log_s \log n$, and we can charge a patching at level i only to these short edges. We then see that a short edge cut by a j'-level ball is only charged for patchings that occur in levels $i \leq j' - \log_s \log n$.

Now, if more than r' short edges cross C, the tour is patched via the minimum spanning tree (à la Lemma 1.8), at a per edge cost of $O\left(\frac{s^i r'^{1-1/\mathrm{ddim}(S)}}{r'/2}\right) = O\left(\frac{s^i\varepsilon}{\mathrm{ddim}(S)}\right)$. Recall though that the edges charged for this patching are edges cut by balls at levels $i + \log_s \log n$ or higher. It follows that the

expected cost to edge (u,v) due to a patching at level i is $O\left(\frac{d(u,v)\,\mathrm{ddim}(S)}{s^{i+\log_s \log n}} \cdot \frac{s^i \varepsilon}{\mathrm{ddim}(S)}\right) = O\left(\frac{\varepsilon d(u,v)}{\log n}\right)$, and due to patchings at all levels is $O(\varepsilon d(u,v))$. This concludes the analysis for the short edges.

We will now argue that the number of long edges entering and exiting C is bounded by $(s/\varepsilon)^{\mathrm{ddim}(S)}$, and this completes the proof of the lemma. Since the tour T is net-respecting, each long edge enters C at an s^ℓ-net point such that $\varepsilon s^{i-1} \le s^\ell \le \varepsilon s^i$. Therefore the long edges enter or exit C at one of $(8s/\varepsilon)^{\mathrm{ddim}(S)}$ candidate portals, and by Lemma 1.10 there are at most $2(8s/\varepsilon)^{\mathrm{ddim}(S)} \le (s/\varepsilon)^{2\,\mathrm{ddim}(S)}$ long edges entering or exiting C. Therefore the total number of entry and exit portals is at most $r := r' + (s/\varepsilon)^{2\,\mathrm{ddim}(S)}$, and so the long edges do not figure into the patchings. □

Let $s = (\log n)^{1/(c''\,\mathrm{ddim}(S))}$ for some constant $c'' \ge 32$. We can now provide an efficient algorithm to find a tour as promised in the last lemma.

Lemma 3.2. *If S admits a net-respecting q-sparse tour T, then there exists a randomized algorithm that, with constant probability, finds a tour T' with $w(T') \le (1+\varepsilon)w(T)$ in time $n^{O(2^{4\,\mathrm{ddim}(S)})} \cdot 2^{O(q \log q (\mathrm{ddim}(S)/\varepsilon)^{4\,\mathrm{ddim}(S)} \sqrt[4]{\log n})}$.*

Proof. If we could compute a hierarchical clustering that satisfies Lemma 3.1, then the standard dynamic program from Section 2 would give a tour for S satisfying Lemma 3.2. However, we cannot compute this hierarchical clustering, since we do not have access to T and cannot know which radii are valid choices for h_u. Instead, we present a dynamic program that guesses the proper value of h_u. Since only a fraction of $\frac{1}{10\,\mathrm{ddim}(S)}$ of the candidate values for h_u are invalid, recall that the exponential distribution of [ABN11] implies that a random guess for the value of h_u is a valid value with probability at least $1/2$. Hence, $O(\log n)$ independent random choices ensure that at least one choice for h_u is valid with probability $1 - \frac{1}{n^2}$, which by a union bound implies that with constant probability, for each net-point at least one of the $O(\log n)$ choices is valid.

The dynamic program guesses a radius for each net-point (i.e., it tries all the $O(\log n)$ random choices made above) and builds the tour in a bottom-up fashion, from level 1 to level L. Recall that $s = (\log n)^{1/c''\,\mathrm{ddim}(S)}$. It follows that the number of levels in the hierarchy is $L = O(\log_s n) = O\left(\frac{\mathrm{ddim}(S)\log n}{\log\log n}\right)$. Suppose by induction that the dynamic program for computing a tour at level $i-1$ has been completed. We show how to compute level i.

Consider some i-level cluster C centered at $u \in H_i$. C is formed by cuts from neighboring balls in levels above i, and we wish to enumerate all possible *formations* of C: Recall that we make $O(\log n)$ random choices for $h_u \in [s^i, 2s^i]$. Further, since u is within distance $4s^i$ of $2^{3\,\mathrm{ddim}(S)}$ other s^i-net points whose radii may cut C, and we guess $O(\log n)$ radii for each of these net-point, C may be cut in $(O(\log n))^{2^{3\,\mathrm{ddim}(S)}}$ different ways in this level. Since C may be cut from above in all $L-i$ levels, it follows that the number of possible formations for C is bounded by $(O(\log n))^{2^{3\,\mathrm{ddim}(S)}L} = n^{O(2^{4\,\mathrm{ddim}(S)})}$. For each fixed formation, we must calculate the minimum (m,r)-light tour for each possible choice of r exit portals of C (m^r possibilities). C has at most $s^{2\,\mathrm{ddim}(S)}$ child clusters, and so there are $(O(\log n))^{2^{3\,\mathrm{ddim}(S)} s^{2\,\mathrm{ddim}(S)}}$ possible child formations. (Note that since the radii of higher level balls has already been fixed, we need not consider all ways that these higher level balls can cut into C's children.). For each fixed child formation, we consider each portal configuration for the set of children ($(m^r)^{s^{2\,\mathrm{ddim}(S)}}$ possibilities). We then compute the cost of connecting the child portals to the r exit portals of C: This can be done by enumerating all graphs with one edge on each vertex, in at most $\left(rs^{2\,\mathrm{ddim}(S)}\right)^{rs^{2\,\mathrm{ddim}(S)}}$ different ways. We can

bound each of the expressions above by:

$$O(mr\log n)^{rs^{2\,\mathrm{ddim}(S)}} = 2^{O(q\log q(\mathrm{ddim}(S)/\varepsilon)^{4\,\mathrm{ddim}(S)}s^{4\,\mathrm{ddim}(S)}(\log\log n)^2)} = 2^{O(q\log q(\mathrm{ddim}(S)/\varepsilon)^{4\,\mathrm{ddim}(S))}\sqrt[4]{\log n})}.$$

Lemma 3.2 follows. □

Comment. For use in Section 4, we note that the previous construction gives an alternate (perhaps stronger) guarantee to that of Lemma 3.2. Let a k-tour of S be a set of k *open* tours that amongst them visit all points of S. The best k-tour is denoted as $\mathrm{kOPT}(S)$. The term (m,r)-light with respect to k-tours is unchanged; each cluster may only be entered or exited via the r portals, once per portal. Similar, a q-sparse k-tour is one in which each i-level ball $B(u,3s^i)$ covers at most qs^i edges of the k-tour. Then the following lemma holds.

Lemma 3.3. *If S admits a net-respecting q-sparse k-tour T, then there exists a randomized algorithm that with constant probability finds a k-tour T' with $w(T') \le (1+\varepsilon)w(T)$ in time $n^{O(2^{4\,\mathrm{ddim}(S)})}\cdot 2^{O(q\log q(\mathrm{ddim}(S)/\varepsilon)^{4\,\mathrm{ddim}(S)}\sqrt[4]{\log n})}$.*

3.2 Eliminating dense areas

Lemmas 3.1 and 3.2 show that sparse tours admit efficient hierarchical decompositions and algorithms. Here, we consider tours that have dense neighborhoods, and show that the point set can be divided into areas with all light tours. We then solve TSP on each subset, and join the resulting subtours into a single tour.

Set $q := (s/\varepsilon)^{O(\mathrm{ddim}(S))}\cdot 2^{O(\mathrm{ddim}^2(S))}$.

Lemma 3.4. *There is a (randomized) polynomial-time algorithm that given a set S (with $|S|>1$), computes two subsets $S_1 \subset S$ and $S_2 \subsetneq S$ with $S_1 \cup S_2 = S$ and $S_1 \cap S_2 \neq \emptyset$, such that*

(a). $\mathrm{OPT}^{NR}(S_1)$ is q'-sparse, for $q' = O(q\sqrt[8]{\log n})$; and

(b). $w(\mathrm{OPT}^{NR}(S_1)) + w(\mathrm{OPT}^{NR}(S_2)) \le w(\mathrm{OPT}^{NR}(S)) + \varepsilon w(\mathrm{OPT}^{NR}(S_1))$.

Proof. The algorithm begins by locating the lowest level i for which there exists $u \in S$ such that $w(\mathrm{MST}(B(u,3s^i)) > 2qs^i$, and setting v to be such that $w(\mathrm{MST}(B(v,3s^i)))$ is maximized. If no such i exists, then we conclude by Lemma 1.12(i) that $\mathrm{OPT}^{NR}(S)$ is $13q$-sparse, and our lemma is trivial: Set $S_1 = S$ and S_2 includes an arbitrary single point. Assume then that i exists, and further that v has been found. Let $q^* := w(\mathrm{MST}(B(v,3s^i))/(2s^i)$, and it follows that $q^* > q$.

Fix a tour $T = \mathrm{OPT}^{NR}(S)$. Ideally, we would now like to choose some radius h, partition S into two point sets $\tilde{S}_1 = B(v,h)$ and $\tilde{S}_2 = S \setminus \tilde{S}_1$, and then show tours for the two sets, whose combined weight is only slightly greater than that of T. Let $\tilde{S}_i^*$ denote the edges of the complete graph on the points of $\tilde{S}_i$ (for $i=1,2$); then we could bound the weight of the subtours by showing that $T_i = \tilde{S}_i^* \cap T$ (for $i=1,2$) can each be made into a closed tour by adding only a light-weight collection of edges to "patch" the edges of T cut by the partition (as in Lemma 1.9). However, this plan may be costly because a radius h ball can cut many edges of T which then need to be patched. Moreover, in order to ensure that the subtours are net-respecting, we need to augment the sets $\tilde{S}_i$ with appropriate (nearby) net points. To solve this problem, we will show how to create sets $S_1 \supset \tilde{S}_1$ (which contains copies of some points of $\tilde{S}_2$) and $S_2 \supset \tilde{S}_2$ (which contains copies of some points of $\tilde{S}_1$) for which the lemma holds.

Below, we will choose (deterministically) a value $h \in [12s^i, 13s^i]$. Assume that h has been chosen, and let us focus on the long edges of T crossing $\tilde{S}_1$, those of length more than δs^i for $\delta = O(\varepsilon/2^{10\,\mathrm{ddim}(S)})$. Since T is net-respecting, these edges must cross $\tilde{S}_1$ at ℓ-level net points, where ℓ is the maximum value such that $s^\ell \le \varepsilon\delta s^i$. We will patch T_1 with respect to these long edges via the minimum spanning tree of the ℓ-level net points covering $\tilde{S}_1$. There are $(s/\varepsilon\delta)^{4\,\mathrm{ddim}(S)}$ such net points, and their net-respecting MST has weight at most $(1+6\varepsilon)(s/\varepsilon\delta)^{4\,\mathrm{ddim}(S)}s^i < \varepsilon q s^i$ (see Lemma 1.11), which bounds, up to a constant factor, the cost of patching the long edges crossing $\tilde{S}_1$.

Turning to the shorter edges of T (of length at most δs^i) crossing $\tilde{S}_1$, we show how to patch T_1 with respect to these edges. Now, these edges cross into $\tilde{S}_1$ from a set of points $V \subset \tilde{S}_2$ within distance $h + \delta s^i$ of v. Since we have shown how to patch the long edges of T_1 crossing ℓ-level net points, the short edges of T_1 crossing V will be patched by connecting them to ℓ'-level net points (though not necessarily directly, see below), where ℓ' is the largest value such that $s^{\ell'} \le \delta s^i$. To this end, add to $\tilde{S}_1$ copies of all ℓ'-level and lower level net points in $\tilde{S}_2$ which cover points in V, and let the resulting point set be S_1. Note that we can now patch the short edges using the MST of the new points, and then convert the new edges to be net-respecting.

We now bound the cost of this patching (which also includes the cost of covering the copies of $\tilde{S}_2$ points in S_1). Define the annulus $A(v, r_1, r_2) = B(v, r_2) \setminus B(v, r_1)$. The patching cost is bounded, up to a constant factor, by the sum of the minimum spanning trees of all balls of radius $s^{\ell'}$ centered at ℓ'-level net points inside the annulus $A(v, h - \delta s^i, h + \delta s^i)$. Let $N(h)$ denote the set of all such net points, and our algorithm chooses the value for h such that the sum $\sum_{u\in N(h)} w(\mathrm{MST}(B(u, 4\ell')))$ is minimum. The number of ℓ'-level net points inside this annulus is upper bounded by $(2\cdot 14s/\delta)^{2\,\mathrm{ddim}(S)} \le (s/\delta)^{4\,\mathrm{ddim}(S)}$. By Lemma 1.12(ii) for each ℓ'-level net point u we have $w(\mathrm{MST}(B(u, s^{\ell'}))) \le w(T \cap B^*(u, 4s^{\ell'})) + (s/\varepsilon)^{2\,\mathrm{ddim}(S)}s^{\ell'}$. Recalling that $s^{\ell'} \le s^i$, the total cost of the patching for short edges is bounded by

$$\sum_{u\in N(h)} w(\mathrm{MST}(B(u, \ell'))) \le \sum_{u\in N(h)} \Big[w(T \cap B^*(u, 4s^{\ell'}))\Big] + (s^6/\varepsilon^2\delta^4)^{\mathrm{ddim}(S)}s^i.$$

Now, each of the balls $B(u, s^{\ell'})$ intersects at most $2^{5\,\mathrm{ddim}(S)}$ other balls centered at ℓ'-level net points. Hence, it follows easily that $\sum_{u\in N(h)} w(T \cap B^*(u, 4s^{\ell'})) \le 2^{5\,\mathrm{ddim}(S)} w(T \cap A^*(v, h - 5\delta s^i, h + 5\delta s^i))$, where $A^*(v, r_1, r_2)$ is defined as all edges with both endpoints inside the annulus $A(v, r_1, r_2)$.

It remains only to show there exists an h giving a low-weight annulus, and this will in turn bound the cost of the patching. By an averaging argument, there is a value for h for which $A^*(v, h - 5\delta s^i, h + 5\delta s^i)$ contains edges of total weight $O(\delta) \cdot w(T \cap B^*(v, 13s^i))$. (Here, we assumed that $12s^i + 5\delta s^i \le h \le 13s^i - 5\delta s^i$.) By Lemma 1.12(i), $w(B^*(v, 13s^i) \cap T) \le 7w(\mathrm{MST}(B(v, 13s^i)))$. To bound $w(\mathrm{MST}(B(v, 13s^i)))$, first recall that v was chosen to maximize $w(\mathrm{MST}(B(v, 3s^i)))$. Now by repeated application of the definition of doubling dimension, $B(v, 13s^i)$ can be covered by $2^{4\,\mathrm{ddim}(S)}$ balls of radius $3s^i$ (with centers from S), and so $\mathrm{MST}(B(v, 13s^i))$ is bounded by the cost of constructing a minimum spanning tree inside each of these small balls and connecting them together. It follows that $w(\mathrm{MST}(B(v, 13s^i)) \le 2^{4\,\mathrm{ddim}(S)} w(\mathrm{MST}(B(v, 3s^i))) + 2^{4\,\mathrm{ddim}(S)} 26s^i \le 2^{5\,\mathrm{ddim}(S)} \cdot 3q^* s^i$. Therefore we obtain

$$\sum_{u\in N(h)} w(T \cap B^*(u, 4s^{\ell'})) \le O(\delta) \cdot 2^{10\,\mathrm{ddim}(S)} q^* s^i.$$

Since $\delta = O(\varepsilon/2^{10\,\mathrm{ddim}(S)})$, this last bound is at most $\varepsilon q^* s^i$. Recalling that $q \ge \varepsilon^{-1}(s^6/\varepsilon^2\delta^4)^{\mathrm{ddim}(S)}$, we can conclude that the total weight of all patchings (including short and long edges) is at most

$O(\varepsilon q s^i) + \varepsilon q^* s^i + (s^6/\varepsilon^2\delta^4)^{\mathrm{ddim}(S)} s^i \leq O(\varepsilon q^* s^i)$. Since $h > 12s^i$, S_1 contains the ball $B(v, 12s^i)$ and so by Lemma 1.12(ii) $w(\mathrm{OPT}^{NR}(S_1)) \geq w(\mathrm{MST}(B(v, 3s^i))) - (s/\varepsilon)^{2\,\mathrm{ddim}(S)} 3s^i = 2q * s^i - (s/\varepsilon)^{2\,\mathrm{ddim}(S)} 3s^i \geq q^* s^i$, so we may conclude that the patching cost is $O(\varepsilon) \cdot w(\mathrm{OPT}^{NR}(S_1))$.

A similar argument applies to the patchings needed for S_2, by adding to $\tilde{S}_2$ all ℓ-level net points of $\tilde{S}_1$, as well as all ℓ'-level net points of $\tilde{S}_1$ covering points of $\tilde{S}_2$; the resulting set is S_2. This completes the proof of part (b) of the lemma.

We note that by construction $S_1 \cap S_2 \neq \emptyset$, and that $S_2 \neq S$ (as implied by part (b) of the lemma).

We now turn to proving part (a). By the choice of i, for every $j < i$ and every net-point $u \in H_j$, $\mathrm{MST}(B(u, 3s^j)) \leq 2qs^j$, and so by Lemma 1.12(i), $w(\mathrm{OPT}^{NR}(S_1) \cap B^*(u, 3s^j)) \leq 14qs^j$. By the packing property, for every net point $u \in H_i$, $w(\mathrm{OPT}^{NR}(S_1) \cap B^*(u, 3s^i)) \leq 14(4s)^{\mathrm{ddim}(S)} qs^{i-1} = O(\sqrt[16]{\log n}) \cdot qs^i$, which completes the lemma.

□

We are now ready to prove Theorem 1.3.

Proof of Theorem 1.3. Given a point set S, if S contains a single point then we are done. Otherwise, we use the procedure of Lemma 3.4 to create two instances of TSP, $S_1 \subseteq S$ and $S_2 \subset S$. S_1 admits a q'-sparse and net-respecting tour T_1 as promised by Lemma 3.4. A tour of almost the same cost (at most $1 + \varepsilon$ factor larger) can be computed by the algorithm of Lemma 3.2, obtaining a tour T_1', where $w(T_1') \leq (1 + \varepsilon)w(T_1)$. Set S_2 is solved inductively (that is S is recursively replaced by S_2), obtaining a tour T_2'. The inequality $S_1 \cap S_2 \neq \emptyset$ implies that separate tours T_1' and T_2' can be joined together to obtain a complete tour T at no additional cost.

We now prove inductively that $w(T) \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right) \cdot w(\mathrm{OPT}^{NR}(S))$. By the induction hypothesis we have that $w(T_2') \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right) \cdot w(\mathrm{OPT}^{NR}(S_2))$. Lemma 3.4 implies that

$$w(T_1') \leq (1+\varepsilon)w(T_1) \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right) \cdot (w(\mathrm{OPT}^{NR}(S)) - w(\mathrm{OPT}^{NR}(S_2))).$$

Therefore

$$w(T) = w(T_1') + w(T_2') \leq w(T_1') + \left(\frac{1+\varepsilon}{1-\varepsilon}\right) \cdot w(\mathrm{OPT}^{NR}(S_2)) \leq \left(\frac{1+\varepsilon}{1-\varepsilon}\right) \cdot w(\mathrm{OPT}^{NR}(S)),$$

proving the inductive claim. Finally, by Lemma 1.11 we have that $w(T) = (1+O(\varepsilon)) \cdot w(OPT(S))$. The runtime follows from executing the algorithm of Lemma 3.2 on the q'-sparse sets of Lemma 3.4. □

4 Spanner dependent algorithm

Here we show that the techniques presented above imply a different TSP dynamic programming algorithm, which has efficient runtime if doubling metrics admit light low-stretch spanners. In particular, if a certain conjecture holds regarding the weight of such a spanner the runtime of the new algorithm has significantly better dependence on the parameters $n, \mathrm{ddim}(S)$ and ε than that of our TSP algorithm described in the main part of the paper.

4.1 Metric spanners

A graph H is a $(1+\varepsilon)$-stretch spanner for graph G if H is a subgraph of G that contains all nodes of G (but not all edges), and $d_H(u,v) \leq (1+\varepsilon)d_G(u,v)$ for all $u, v \in G$, where $d_G(u,v)$ ($d_H(u,v)$) denotes the shortest path distance between u and v on G (H).

Let G be the full graph of an arbitrary d-dimensional Euclidean set S. Then there exists a $(1+\varepsilon)$-stretch spanner for G with weight $W_E \operatorname{MST}(S)$, where $W_E = \varepsilon^{-\Theta(d)}$ [DNS95, ADM+95], and the bound on W_E is tight. Now let G be the full graph of a doubling set S. Then there exists a $(1+\varepsilon)$-stretch spanner for G with weight $W_D \operatorname{MST}(S)$, where $W_D = \varepsilon^{-\Theta(\operatorname{ddim}(S))} \log n$ [Smi09], though it is not known if this upper bound on W_D is tight. (Of course, the lower bound for Euclidean spanners also applies to the more general doubling spanners.) In fact, the following conjecture has been the subject of much study in the spanner community:

Conjecture 1. *W_D is independent of n; perhaps $W_D = \varepsilon^{-\Theta(\operatorname{ddim}(S))}$.*

We will give an algorithm for TSP whose runtime depends on W_D.

4.2 Hierarchy and algorithms

In this section, we briefly sketch a proof of the following theorem:

Theorem 4.1. *There exists a randomized algorithm that with constant probability computes a $(1+\varepsilon)$-approximation to an optimal TSP metric tour in time $n\,(\log n)^{(W_D 2^{\operatorname{ddim}(S)}/\varepsilon)^{O(\operatorname{ddim}(S))} \log\log n}$.*

The proof proceeds along the lines of the proof for Theorem 1.3, but makes use of the doubling spanner G. (As an aside, we may assumes that all edges of G are unit length, and this construction can be attained via the work of [GT08].) Let $s = O(1)$. If graph G is q-sparse, then we use the standard TSP dynamic programming algorithm to produce a $(1+\varepsilon W_D)$-approximation the optimal tour on G. However, we need not guess the valid radii for a net-point; instead, we choose a random radius from among those that cut at most $\operatorname{ddim}(S)q$ edges of G. We can then run the standard dynamic programming algorithm of [Aro98, Tal04] in time m^r, with m and r as in Lemma 3.1.

To handle the removal of dense subsets, we find directly the lowest level i that contains a dense ball. Let $B(u, 3s^i)$ be the heaviest such ball in i. We choose the radius $h_u \in [3s^i, 4s^i]$ which cuts the fewest edges of G (at most $\operatorname{ddim}(S)q$). Let S_1 be the points inside $B(u, h_u)$. We remove S_1 from S, find a $\operatorname{ddim}(S)q$-tour for S_1 using Lemma 3.3, and then convert this into a closed tour by adding edges of the minimum spanning tree of the cross-points. The second set S_2 consists of $S - S_1$, plus copies of the points of the previous minimum spanning tree. The analysis is similar to that in Section 3.2, and shows that the weight of the minimum spanning tree is much lighter than $w(\operatorname{OPT}(S_1))$. The final analysis gives that the weight of computed tour is $\operatorname{OPT}(S) + \varepsilon w(G) = (1+\varepsilon W_D)\operatorname{OPT}(S)$, and Theorem 4.1 follows by scaling ε to $\frac{\varepsilon}{W_D}$.

Acknowledgements. We thank Ittai Abraham, Alex Andoni, Anupam Gupta, Liam Roditty and Kunal Talwar for helpful discussions.

References

[ABCC07] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook. *The Traveling Salesman Problem: A Computational Study (Princeton Series in Applied Mathematics).* Princeton University Press, Princeton, NJ, USA, 2007.

[ABN08] I. Abraham, Y. Bartal, and O. Neiman. Embedding metric spaces in their intrinsic dimension. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 363–372, 2008.

[ABN11] I. Abraham, Y. Bartal, and O. Neiman. Advances in metric embedding theory. *Advances in Mathematics*, 228(6):3026 – 3126, 2011.

[ACGP10] I. Abraham, S. Chechik, C. Gavoille, and D. Peleg. Forbidden-set distance labels for graphs of bounded doubling dimension. In *29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pages 192–200. ACM, 2010.

[ADM+95] S. Arya, G. Das, D. M. Mount, J. S. Salowe, and M. Smid. Euclidean spanners: Short, thin, and lanky. In *27th annual ACM symposium on Theory of computing*, pages 489–498, 1995.

[Aro98] S. Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *J. ACM*, 45:753–782, 1998.

[ARR98] S. Arora, P. Raghavan, and S. Rao. Approximation schemes for euclidean k-medians and related problems. In *30th annual ACM symposium on Theory of computing*, pages 106–113. ACM, 1998.

[Ass83] P. Assouad. Plongements lipschitziens dans $\mathbf{R}^n$. *Bull. Soc. Math. France*, 111(4):429–448, 1983.

[Bar96] Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science*, pages 184–193. IEEE Computer Society, 1996.

[Bar98] Y. Bartal. On approximating arbitrary metrices by tree metrics. In *30th annual ACM symposium on Theory of computing*, pages 161–168. ACM, 1998.

[CE11] T.-H. H. Chan and K. M. Elbassioni. A QPTAS for TSP with fat weakly disjoint neighborhoods in doubling metrics. *Discrete & Computational Geometry*, 46(4):704–723, 2011.

[CG08] T.-H. H. Chan and A. Gupta. Approximating TSP on metrics with bounded global growth. In *19th annual ACM-SIAM symposium on Discrete algorithms*, pages 690–699. SIAM, 2008.

[Chr76] N. Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie-Mellon Univ. Management Sciences Research Group, 1976.

[CL98] A. Czumaj and A. Lingas. A polynomial time approximation scheme for euclidean minimum cost k-connectivity. In *25th International Colloquium on Automata, Languages and Programming*, ICALP '98, pages 682–694. Springer-Verlag, 1998.

[Cla99] K. L. Clarkson. Nearest neighbor queries in metric spaces. *Discrete Comput. Geom.*, 22(1):63–93, 1999.

[CLZ02] A. Czumaj, A. Lingas, and H. Zhao. Polynomial-time approximation schemes for the euclidean survivable network design problem. In *29th International Colloquium on Automata, Languages and Programming*, ICALP '02, pages 973–984. Springer-Verlag, 2002.

[DNS95] G. Das, G. Narasimhan, and J. Salowe. A new way to weigh malnourished euclidean graphs. In *6th annual ACM-SIAM symposium on Discrete algorithms*, pages 215–222. SIAM, 1995.

[FRT03] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *35th annual ACM symposium on Theory of computing*, pages 448–455. ACM, 2003.

[GGN06] J. Gao, L. J. Guibas, and A. Nguyen. Deformable spanners and applications. *Comput. Geom. Theory Appl.*, 35(1), 2006.

[GKK10] L.-A. Gottlieb, L. Kontorovich, and R. Krauthgamer. Efficient classification for metric data. In *23rd Conference on Learning Theory*, pages 433–440. Omnipress, 2010.

[GKL03] A. Gupta, R. Krauthgamer, and J. R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *44th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '03, pages 534–543. IEEE Computer Society, 2003.

[GP02] G. Gutin and A. P. Punnen, editors. *The traveling salesman problem and its variations*, volume 12 of *Combinatorial Optimization*. Kluwer Academic Publishers, Dordrecht, 2002.

[GT08] A. Gupta and K. Talwar. How to complete a doubling metric. In *8th Latin American conference onTheoretical informatics*, pages 36–47. Springer-Verlag, 2008.

[Kar72] R. Karp. Reducibility among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*. Plenum Press, 1972.

[KL04] R. Krauthgamer and J. R. Lee. Navigating nets: Simple algorithms for proximity search. In *15th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 791–801, January 2004.

[KR07] S. G. Kolliopoulos and S. Rao. A nearly linear-time approximation scheme for the Euclidean k-median problem. *SIAM J. Comput.*, 37:757–782, June 2007.

[Laa00] T. J. Laakso. Ahlfors Q-regular spaces with arbitrary $Q > 1$ admitting weak Poincaré inequality. *Geom. Funct. Anal.*, 10(1):111–123, 2000.

[Laa02] T. J. Laakso. Plane with A_∞-weighted metric not bi-Lipschitz embeddable to $\mathbb{R}^N$. *Bull. London Math. Soc.*, 34(6):667–676, 2002.

[Lee10] J. R. Lee. The Gödel prize, TSP, and volume growth (blog post). `http://tcsmath.wordpress.com/2010/06/24/the-godel-prize-tsp-and-volume-growth/`, June 2010.

[LLKS85] E. L. Lawler, J. K. Lenstra, A. H. G. R. Kan, and D. B. Shmoys, editors. *The Traveling Salesman Problem*. Wiley-Interscience series in discrete mathematics, 1985.

[LP01] U. Lang and C. Plaut. Bilipschitz embeddings of metric spaces into space forms. *Geom. Dedicata*, 87(1-3):285–307, 2001.

[Mit99] J. S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. *SIAM J. Comput.*, 28(4):1298–1309, 1999.

[Mit07] J. S. B. Mitchell. A PTAS for TSP with neighborhoods among fat regions in the plane. In *8th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 11–18. SIAM, 2007.

[Pap77] C. H. Papadimitriou. The Euclidean travelling salesman problem is NP-complete. *Theoretical Computer Science*, 4(3):237 – 244, 1977.

[PV06] C. Papadimitriou and S. Vempala. On the approximability of the traveling salesman problem. *Combinatorica*, 26:101–120, 2006.

[PY93] C. H. Papadimitriou and M. Yannakakis. The traveling salesman problem with distances one and two. *Math. Oper. Res.*, 18:1–11, February 1993.

[Rei94] G. Reinelt. *The Traveling Salesman*, volume 840 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1994.

[RS98] S. B. Rao and W. D. Smith. Approximating geometrical graphs via "spanners" and "banyans". In *30th annual ACM symposium on Theory of computing*, pages 540–550. ACM, 1998.

[Smi09] M. Smid. The weak gap property in metric spaces of bounded doubling dimension. In *Efficient Algorithms: Essays Dedicated to Kurt Mehlhorn on the Occasion of his 60th Birthday*, pages 275–289. Springer-Verlag, 2009.

[Smi10] M. Smid. On some combinatorial problems in metric spaces of bounded doubling dimension. Manuscript, available at `http://people.scs.carleton.ca/~michiel/research.html`, 2010.

[Tal04] K. Talwar. Bypassing the embedding: algorithms for low dimensional metrics. In *36th annual ACM symposium on Theory of computing*, pages 281–290. ACM, 2004.

[Tre00] L. Trevisan. When Hamming meets Euclid: The approximability of geometric TSP and Steiner tree. *SIAM Journal on Computing*, 30(2):475–485, 2000.